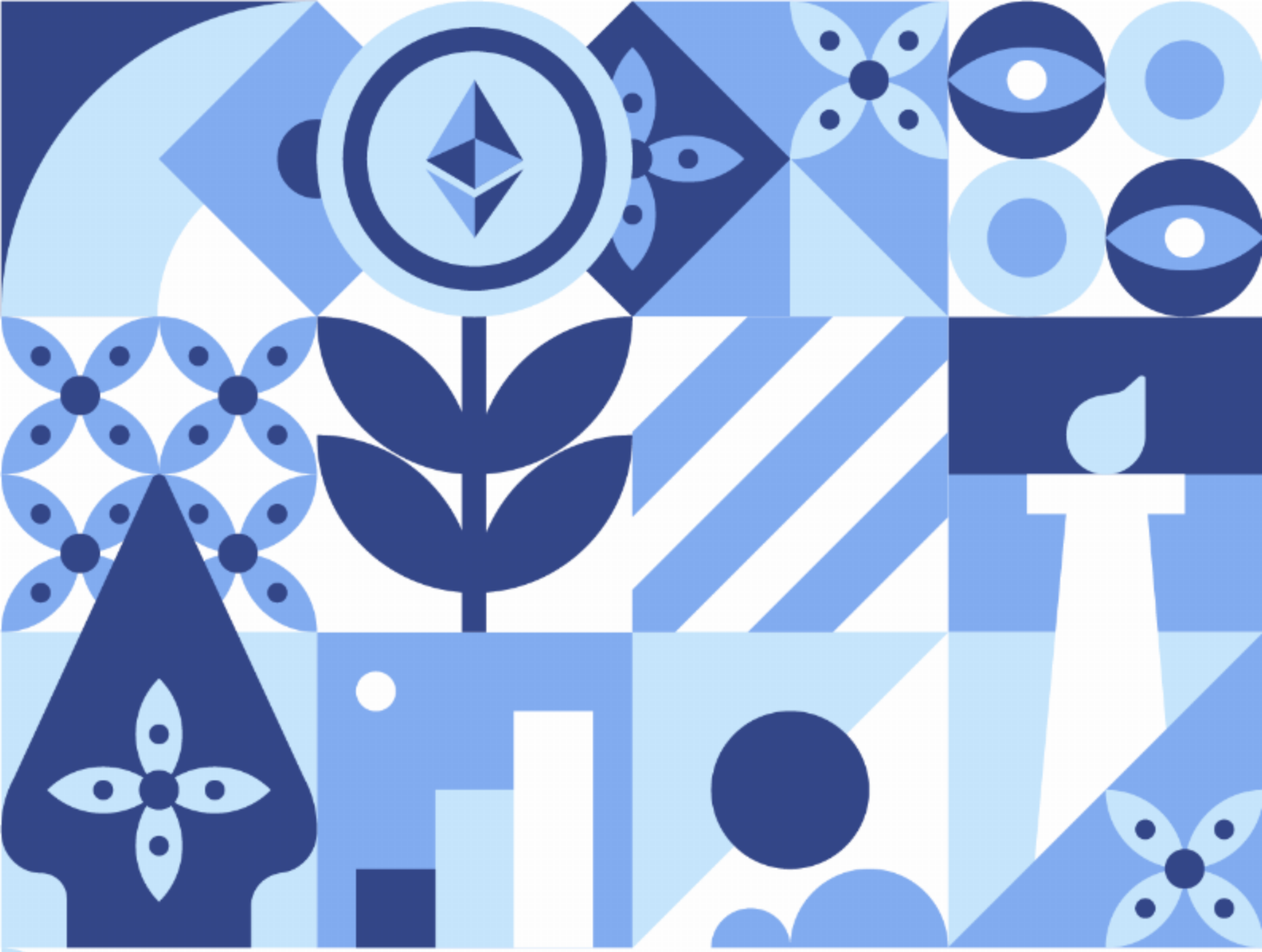

BunnyVerse

Smart Contract Audit Report

SYSFIXED

21 Dec 2022

TABLE OF CONTENTS

AUDITED DETAILS

Audited Project

Project name	Token ticker	Blockchain
BunnyVerse	BNV	Ethereum

Addresses

Contract address	0x072987D5B36aD8d45552aEd98879a7101cCdd749
Contract deployer address	0x1578265d37E4abDAeBA400674ad4f720439F7c79

Project Website

https://bunny-verse.com/

Codebase

https://etherscan.io/address/0x072987D5B36aD8d45552aEd98879a7101cCdd749#code

SUMMARY

BunnyVerse (BNV) is more than just a meme. It is an ERC 20 token with the actual utility connected to it. The BunnyVerse team has the ambition to create its ecosystem. Our ecosystem will work hard to deliver the best products within the crypto, web3 and metaverse space. The BunnyVerse will be a platform and launchpad for newly developed and released games targeting sophisticated gaming audiences.

Contract Summary

Documentation Quality

BunnyVerse provides a very good documentation with standard of solidity base code.

- The technical description is provided clearly and structured and also dont have any high risk issue.

Code Quality

The Overall quality of the basecode is standard.

- Standard solidity basecode and rules are already followed by BunnyVerse with the discovery of several low issues.

Test Coverage

Test coverage of the project is 100% (Through Codebase)

Audit Findings Summary

- SWC-100 SWC-108 | Explicitly define visibility for all state variables on lines 682.
- SWC-101 | It is recommended to use vetted safe math libraries for arithmetic operations consistently on lines 50, 66, 76, 77, 92, 108, 619, 619, 620, 620, 668, 719, 719, 720, 720, 721, 721, 874, 903, 932, 963, 978, 980, 1024, 1044, 1051, 1079, 1087, 1104, 1104, 1110, 1110, 1115, 1140, 1144, 1144, 1144, 1155, 1378, 1392, 1392, 1392, 1393, 1393, 1393, 1395, 1395, 1395, 1396, 1396, 1396, 1406, 1414, 1451, 1451, 1461, 1461 and 980.
- SWC-110 SWC-123 | It is recommended to use of revert(), assert(), and require() in Solidity, and the new REVERT opcode in the EVM on lines 875, 933, 934, 979, 980, 980, 1171, 1172, 1380, 1381, 1383, 1384, 1483 and 1484.
- SWC-115 | tx.origin should not be used for authorization, use msg.sender instead on lines 1036 and 1037.
- SWC-120 | It is recommended to use external sources of randomness via oracles on lines 865, 1024, 1036 and 1037.

CONCLUSION

We have audited the BunnyVerse project released on December 2022 to discover issues and identify potential security vulnerabilities in BunnyVerse Project. This process is used to find technical issues and security loopholes which might be found in the smart contract.

The security audit report provides a satisfactory result with some low-risk issues.

The issues found in the BunnyVerse smart contract code do not pose a considerable risk. The writing of the contract is close to the standard of writing contracts in general. The low-risk issues found are some arithmetic operation issues, a state variable visibility is not set, weak sources of randomness, tx.origin as a part of authorization control and out of bounds array access which the index access expression can cause an exception in case of the use of an invalid array index value. We recommend avoiding using " "tx.origin". The tx.origin environment variable has been found to influence a control flow decision. Note that using "tx.origin" as a security control might cause a situation where a user inadvertently authorizes a smart contract to perform an action on their behalf. It is recommended to use "msg.sender" instead.

AUDIT RESULT

Article	Category	Description	Result
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	ISSUE FOUND
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	ISSUE FOUND
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	PASS
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	PASS
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	PASS
Unprotected Ether Withdrawal	SWC-105	Due to missing or insufficient access controls, malicious parties can withdraw from the contract.	PASS
SELFDESTRUCT Instruction	SWC-106	The contract should not be self-destructible while it has funds belonging to users.	PASS
Reentrancy	SWC-107	Check effect interaction pattern should be followed if the code performs recursive call.	PASS
Uninitialized Storage Pointer	SWC-109	Uninitialized local storage variables can point to unexpected storage locations in the contract.	PASS
Assert Violation	SWC-110 SWC-123	Properly functioning code should never reach a failing assert statement.	ISSUE FOUND
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	PASS
Delegate call to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	PASS

DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless required.	PASS
Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	PASS
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	ISSUE FOUND
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	PASS
Signature Unique ID	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	PASS
Incorrect Constructor Name	SWC-118	Constructors are special functions that are called only once during the contract creation.	PASS
Shadowing State Variable	SWC-119	State variables should not be shadowed.	PASS
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes or be predictable.	ISSUE FOUND
Write to Arbitrary Storage Location	SWC-124	The contract is responsible for ensuring that only authorized user or contract accounts may write to sensitive storage locations.	PASS
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order. The rule of thumb is to inherit contracts from more /general/ to more /specific/.	PASS
Insufficient Gas Griefing	SWC-126	Insufficient gas griefing attacks can be performed on contracts which accept data and use it in a sub-call on another contract.	PASS
Arbitrary Jump Function	SWC-127	As Solidity doesnt support pointer arithmetics, it is impossible to change such variable to an arbitrary value.	PASS

Typographical Error	SWC-129	A typographical error can occur for example when the intent of a defined operation is to sum a number to a variable.	PASS
Override control character	SWC-130	Malicious actors can use the Right-To-Left-Override unicode character to force RTL text rendering and confuse users as to the real intent of a contract.	PASS
Unused variables	SWC-131 SWC-135	Unused variables are allowed in Solidity and they do not pose a direct security issue.	PASS
Unexpected Ether balance	SWC-132	Contracts can behave erroneously when they strictly assume a specific Ether balance.	PASS
Hash Collisions Variable	SWC-133	Using abi.encodePacked() with multiple variable length arguments can, in certain situations, lead to a hash collision.	PASS
Hardcoded gas amount	SWC-134	The transfer() and send() functions forward a fixed amount of 2300 gas.	PASS
Unencrypted Private Data	SWC-136	It is a common misconception that private type variables cannot be read.	PASS

SMART CONTRACT ANALYSIS

Started	Tuesday Dec 20 2022 19:41:40 GMT+0000 (Coordinated Universal Time)
Finished	Wednesday Dec 21 2022 02:00:16 GMT+0000 (Coordinated Universal Time)
Mode	Standard
Main Source File	BUNNYVERSE.sol

Detected Issues

ID	Title	Severity	Status
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "%" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged

SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "-" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "++" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+=" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "/" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "*" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "**" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	ARITHMETIC OPERATION "+" DISCOVERED	low	acknowledged
SWC-101	COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED	low	acknowledged

SWC-108	STATE VARIABLE VISIBILITY IS NOT SET.	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-115	USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-110	OUT OF BOUNDS ARRAY ACCESS	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged
SWC-120	POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.	low	acknowledged

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 50

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
49   function add(uint256 a, uint256 b) internal pure returns (uint256) {
50   uint256 c = a + b;
51   require(c >= a, "SafeMath: addition overflow");
52
53   return c;
54
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 66

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
65   require(b <= a, errorMessage);
66   uint256 c = a - b;
67
68   return c;
69   }
70
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 76

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
75
76   uint256 c = a * b;
77   require(c / a == b, "SafeMath: multiplication overflow");
78
79   return c;
80
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 77

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
76   uint256 c = a * b;
77   require(c / a == b, "SafeMath: multiplication overflow");
78
79   return c;
80   }
81
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 92

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
91   require(b > 0, errorMessage);
92   uint256 c = a / b;
93   // assert(a == b * c + a % b); // There is no case in which this doesn't hold
94
95   return c;
96
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 108

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
107   require(b != 0, errorMessage);
108   return a % b;
109   }
110   }
111
112
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 619

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
618   uint256 private constant MAX = ~uint256(0);
619   uint256 private constant _tTotal = 1 * 1e12 * 1e18;
620   uint256 private _rTotal = (MAX - (MAX % _tTotal));
621   uint256 private _tFeeTotal;
622
623
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 619

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
618   uint256 private constant MAX = ~uint256(0);
619   uint256 private constant _tTotal = 1 * 1e12 * 1e18;
620   uint256 private _rTotal = (MAX - (MAX % _tTotal));
621   uint256 private _tFeeTotal;
622
623
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 620

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
619   uint256 private constant _tTotal = 1 * 1e12 * 1e18;
620   uint256 private _rTotal = (MAX - (MAX % _tTotal));
621   uint256 private _tFeeTotal;
622
623   string private constant _name = "BunnyVerse";
624
```

SWC-101 | ARITHMETIC OPERATION "%" DISCOVERED

LINE 620

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
619   uint256 private constant _tTotal = 1 * 1e12 * 1e18;
620   uint256 private _rTotal = (MAX - (MAX % _tTotal));
621   uint256 private _tFeeTotal;
622
623   string private constant _name = "BunnyVerse";
624
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 668

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
667   bool private gasLimitActive = true;
668   uint256 private gasPriceLimit = 602 * 1 gwei;
669
670   // store addresses that a automatic market maker pairs. Any transfer *to* these
addresses
671   // could be subject to a maximum transfer amount
672
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 719

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
718
719   maxTransactionAmount = _tTotal * 50 / 10000; // 0.5% max txn
720   minimumTokensBeforeSwap = _tTotal * 5 / 10000; // 0.05%
721   maxWallet = _tTotal * 100 / 10000; // 1%
722
723
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 719

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
718
719   maxTransactionAmount = _tTotal * 50 / 10000; // 0.5% max txn
720   minimumTokensBeforeSwap = _tTotal * 5 / 10000; // 0.05%
721   maxWallet = _tTotal * 100 / 10000; // 1%
722
723
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 720

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
719   maxTransactionAmount = _tTotal * 50 / 10000; // 0.5% max txn
720   minimumTokensBeforeSwap = _tTotal * 5 / 10000; // 0.05%
721   maxWallet = _tTotal * 100 / 10000; // 1%
722
723   _rOwned[newOwner] = _rTotal;
724
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 720

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
719   maxTransactionAmount = _tTotal * 50 / 10000; // 0.5% max txn
720   minimumTokensBeforeSwap = _tTotal * 5 / 10000; // 0.05%
721   maxWallet = _tTotal * 100 / 10000; // 1%
722
723   _rOwned[newOwner] = _rTotal;
724
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 721

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
720   minimumTokensBeforeSwap = _tTotal * 5 / 10000; // 0.05%
721   maxWallet = _tTotal * 100 / 10000; // 1%
722
723   _rOwned[newOwner] = _rTotal;
724
725
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 721

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
720   minimumTokensBeforeSwap = _tTotal * 5 / 10000; // 0.05%
721   maxWallet = _tTotal * 100 / 10000; // 1%
722
723   _rOwned[newOwner] = _rTotal;
724
725
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 874

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
873   function manageSnipers(address[] calldata addresses, bool status) public onlyOwner
{
874   for (uint256 i; i < addresses.length; ++i) {
875   _isSniper[addresses[i]] = status;
876   }
877   }
878
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 903

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
902   require(gas >= 300);
903   gasPriceLimit = gas * 1 gwei;
904   }
905
906   // disable Transfer delay
907
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 932

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
931   buyOrSellSwitch = TRANSFER;
932   for(uint256 i = 0; i < airdropWallets.length; i++){
933   address wallet = airdropWallets[i];
934   uint256 airdropAmount = amount[i];
935   _tokenTransfer(msg.sender, wallet, airdropAmount);
936
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 963

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
962   require(!_isExcluded[account], "Account is already excluded");
963   require(_excluded.length + 1 <= 50, "Cannot exclude more than 50 accounts.  Include
a previously excluded address.");
964   if (_rOwned[account] > 0) {
965   _tOwned[account] = tokenFromReflection(_rOwned[account]);
966   }
967
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 978

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
977   require(_isExcluded[account], "Account is not excluded");
978   for (uint256 i = 0; i < _excluded.length; i++) {
979   if (_excluded[i] == account) {
980   _excluded[i] = _excluded[_excluded.length - 1];
981   _tOwned[account] = 0;
982
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 980

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
979   if (_excluded[i] == account) {
980   _excluded[i] = _excluded[_excluded.length - 1];
981   _tOwned[account] = 0;
982   _isExcluded[account] = false;
983   _excluded.pop();
984
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1024

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1023   ){
1024   if(tradingActiveBlock > 0 && (tradingActiveBlock + deadBlocks) > block.number){
1025   _isSniper[to]=true;
1026   }
1027
1028
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1044

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1043   require(amount <= maxTransactionAmount, "Buy transfer amount exceeds the
maxTransactionAmount.");
1044   require(amount + balanceOf(to) <= maxWallet, "Cannot exceed max wallet");
1045   }
1046   //when sell
1047   else if (automatedMarketMakerPairs[to] && !_isExcludedMaxTransactionAmount[from])
{
1048
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1051

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1050   else if (!_isExcludedMaxTransactionAmount[to]){
1051   require(amount + balanceOf(to) <= maxWallet, "Cannot exceed max wallet");
1052   }
1053   }
1054   }
1055
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1079

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1078   _taxFee = _buyTaxFee;
1079   _liquidityFee = _buyLiquidityFee + _buyMarketingFee;
1080   if(_liquidityFee > 0){
1081   buyOrSellSwitch = BUY;
1082   }
1083
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1087

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1086   _taxFee = _sellTaxFee;
1087   _liquidityFee = _sellLiquidityFee + _sellMarketingFee;
1088   if(_liquidityFee > 0){
1089   buyOrSellSwitch = SELL;
1090   }
1091
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1104

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1103   require(percent <= 50, "Swap amount cannot be higher than 0.5% total supply.");
1104   minimumTokensBeforeSwap = _tTotal * percent / 10000;
1105   return true;
1106   }
1107
1108
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1104

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1103   require(percent <= 50, "Swap amount cannot be higher than 0.5% total supply.");
1104   minimumTokensBeforeSwap = _tTotal * percent / 10000;
1105   return true;
1106   }
1107
1108
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1110

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1109   require(percent >= 10, "Cannot set maxTransactionAmount lower than 0.1%");
1110   maxTransactionAmount = _tTotal * percent / 10000;
1111   }
1112
1113   // percent 25 for .25%
1114
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1110

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1109   require(percent >= 10, "Cannot set maxTransactionAmount lower than 0.1%");
1110   maxTransactionAmount = _tTotal * percent / 10000;
1111   }
1112
1113   // percent 25 for .25%
1114
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1115

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1114   function manualBurnLiquidityPairTokens(uint256 percent) external onlyOwner returns
(bool){
1115   require(block.timestamp > lastManualLpBurnTime + manualBurnFrequency , "Must wait
for cooldown to finish");
1116   require(percent <= 1000, "May not nuke more than 10% of tokens in LP");
1117   lastManualLpBurnTime = block.timestamp;
1118
1119
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1140

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1139   bool success;
1140   uint256 totalTokensToSwap = _liquidityTokensToSwap + _marketingTokensToSwap;
1141   if(totalTokensToSwap == 0 || contractBalance == 0) {return;}
1142
1143   // Halve the amount of liquidity tokens
1144
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1144

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1143   // Halve the amount of liquidity tokens
1144   uint256 tokensForLiquidity = (contractBalance * _liquidityTokensToSwap /
totalTokensToSwap) / 2;
1145   uint256 amountToSwapForBNB = contractBalance.sub(tokensForLiquidity);
1146
1147   uint256 initialBNBBalance = address(this).balance;
1148
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1144

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1143   // Halve the amount of liquidity tokens
1144   uint256 tokensForLiquidity = (contractBalance * _liquidityTokensToSwap /
totalTokensToSwap) / 2;
1145   uint256 amountToSwapForBNB = contractBalance.sub(tokensForLiquidity);
1146
1147   uint256 initialBNBBalance = address(this).balance;
1148
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1144

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1143   // Halve the amount of liquidity tokens
1144   uint256 tokensForLiquidity = (contractBalance * _liquidityTokensToSwap /
totalTokensToSwap) / 2;
1145   uint256 amountToSwapForBNB = contractBalance.sub(tokensForLiquidity);
1146
1147   uint256 initialBNBBalance = address(this).balance;
1148
```

SWC-101 | ARITHMETIC OPERATION "-" DISCOVERED

LINE 1155

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1154
1155   uint256 bnbForLiquidity = bnbBalance - bnbForMarketing;
1156
1157   _liquidityTokensToSwap = 0;
1158   _marketingTokensToSwap = 0;
1159
```

SWC-101 | ARITHMETIC OPERATION "++" DISCOVERED

LINE 1378

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1377   uint256 tSupply = _tTotal;
1378   for (uint256 i = 0; i < _excluded.length; i++) {
1379   if (
1380   _rOwned[_excluded[i]] > rSupply ||
1381   _tOwned[_excluded[i]] > tSupply
1382
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1392

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1391   if(buyOrSellSwitch == BUY){
1392   _liquidityTokensToSwap += tLiquidity * _buyLiquidityFee / _liquidityFee;
1393   _marketingTokensToSwap += tLiquidity * _buyMarketingFee / _liquidityFee;
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1392

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1391   if(buyOrSellSwitch == BUY){
1392   _liquidityTokensToSwap += tLiquidity * _buyLiquidityFee / _liquidityFee;
1393   _marketingTokensToSwap += tLiquidity * _buyMarketingFee / _liquidityFee;
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1392

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1391   if(buyOrSellSwitch == BUY){
1392   _liquidityTokensToSwap += tLiquidity * _buyLiquidityFee / _liquidityFee;
1393   _marketingTokensToSwap += tLiquidity * _buyMarketingFee / _liquidityFee;
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1393

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1392   _liquidityTokensToSwap += tLiquidity * _buyLiquidityFee / _liquidityFee;
1393   _marketingTokensToSwap += tLiquidity * _buyMarketingFee / _liquidityFee;
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1393

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1392   _liquidityTokensToSwap += tLiquidity * _buyLiquidityFee / _liquidityFee;
1393   _marketingTokensToSwap += tLiquidity * _buyMarketingFee / _liquidityFee;
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1393

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1392   _liquidityTokensToSwap += tLiquidity * _buyLiquidityFee / _liquidityFee;
1393   _marketingTokensToSwap += tLiquidity * _buyMarketingFee / _liquidityFee;
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1395

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397   }
1398   uint256 currentRate = _getRate();
1399
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1395

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397   }
1398   uint256 currentRate = _getRate();
1399
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1395

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1394   } else if(buyOrSellSwitch == SELL){
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397   }
1398   uint256 currentRate = _getRate();
1399
```

SWC-101 | ARITHMETIC OPERATION "+=" DISCOVERED

LINE 1396

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397   }
1398   uint256 currentRate = _getRate();
1399   uint256 rLiquidity = tLiquidity.mul(currentRate);
1400
```

SWC-101 | ARITHMETIC OPERATION "/" DISCOVERED

LINE 1396

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397   }
1398   uint256 currentRate = _getRate();
1399   uint256 rLiquidity = tLiquidity.mul(currentRate);
1400
```

SWC-101 | ARITHMETIC OPERATION "*" DISCOVERED

LINE 1396

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1395   _liquidityTokensToSwap += tLiquidity * _sellLiquidityFee / _liquidityFee;
1396   _marketingTokensToSwap += tLiquidity * _sellMarketingFee / _liquidityFee;
1397   }
1398   uint256 currentRate = _getRate();
1399   uint256 rLiquidity = tLiquidity.mul(currentRate);
1400
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1406

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1405   function calculateTaxFee(uint256 _amount) private view returns (uint256) {
1406   return _amount.mul(_taxFee).div(10**2);
1407   }
1408
1409   function calculateLiquidityFee(uint256 _amount)
1410
```

SWC-101 | ARITHMETIC OPERATION "**" DISCOVERED

LINE 1414

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1413   {
1414   return _amount.mul(_liquidityFee).div(10**2);
1415   }
1416
1417   function removeAllFee() private {
1418
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1451

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1450   _buyMarketingFee = buyMarketingFee;
1451   require(_buyTaxFee + _buyLiquidityFee + _buyMarketingFee <= 15, "Must keep taxes
below 15%");
1452   }
1453
1454   function setSellFee(uint256 sellTaxFee, uint256 sellLiquidityFee, uint256
sellMarketingFee)
1455
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1451

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1450   _buyMarketingFee = buyMarketingFee;
1451   require(_buyTaxFee + _buyLiquidityFee + _buyMarketingFee <= 15, "Must keep taxes
below 15%");
1452   }
1453
1454   function setSellFee(uint256 sellTaxFee, uint256 sellLiquidityFee, uint256
sellMarketingFee)
1455
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1461

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1460   _sellMarketingFee = sellMarketingFee;
1461   require(_sellTaxFee + _sellLiquidityFee + _sellMarketingFee <= 25, "Must keep
taxes below 25%");
1462   }
1463
1464   function setMarketingAddress(address _marketingAddress) external onlyOwner {
1465
```

SWC-101 | ARITHMETIC OPERATION "+" DISCOVERED

LINE 1461

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
1460   _sellMarketingFee = sellMarketingFee;
1461   require(_sellTaxFee + _sellLiquidityFee + _sellMarketingFee <= 25, "Must keep
taxes below 25%");
1462   }
1463
1464   function setMarketingAddress(address _marketingAddress) external onlyOwner {
1465
```

SWC-101 | COMPILER-REWRITABLE "<UINT> - 1" DISCOVERED

LINE 980

low SEVERITY

This plugin produces issues to support false positive discovery within mythril.

Source File

- BUNNYVERSE.sol

Locations

```
979   if (_excluded[i] == account) {
980   _excluded[i] = _excluded[_excluded.length - 1];
981   _tOwned[account] = 0;
982   _isExcluded[account] = false;
983   _excluded.pop();
984
```

SWC-108 | STATE VARIABLE VISIBILITY IS NOT SET.

LINE 682

low SEVERITY

It is best practice to set the visibility of state variables explicitly. The default visibility for "inSwapAndLiquify" is internal. Other possible visibility settings are public and private.

Source File

- BUNNYVERSE.sol

Locations

```
681
682   bool inSwapAndLiquify;
683   bool public swapAndLiquifyEnabled = false;
684
685   event RewardLiquidityProviders(uint256 tokenAmount);
686
```

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 1036

low SEVERITY

The tx.origin environment variable has been found to influence a control flow decision. Note that using "tx.origin" as a security control might cause a situation where a user inadvertently authorizes a smart contract to perform an action on their behalf. It is recommended to use "msg.sender" instead.

Source File

- BUNNYVERSE.sol

Locations

```
1035   if (to != owner() && to != address(uniswapV2Router) && to !=
address(uniswapV2Pair)){
1036   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1037   _holderLastTransferTimestamp[tx.origin] = block.number;
1038   }
1039   }
1040
```

SWC-115 | USE OF "TX.ORIGIN" AS A PART OF AUTHORIZATION CONTROL.

LINE 1037

low SEVERITY

Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" unless you really know what you are doing.

Source File

- BUNNYVERSE.sol

Locations

```
1036   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1037   _holderLastTransferTimestamp[tx.origin] = block.number;
1038   }
1039   }
1040
1041
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 875

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
874   for (uint256 i; i < addresses.length; ++i) {
875   _isSniper[addresses[i]] = status;
876   }
877   }
878
879
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 933

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
932   for(uint256 i = 0; i < airdropWallets.length; i++){
933   address wallet = airdropWallets[i];
934   uint256 airdropAmount = amount[i];
935   _tokenTransfer(msg.sender, wallet, airdropAmount);
936   }
937
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 934

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
933   address wallet = airdropWallets[i];
934   uint256 airdropAmount = amount[i];
935   _tokenTransfer(msg.sender, wallet, airdropAmount);
936   }
937   restoreAllFee();
938
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 979

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
978   for (uint256 i = 0; i < _excluded.length; i++) {
979   if (_excluded[i] == account) {
980   _excluded[i] = _excluded[_excluded.length - 1];
981   _tOwned[account] = 0;
982   _isExcluded[account] = false;
983
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 980

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
979   if (_excluded[i] == account) {
980   _excluded[i] = _excluded[_excluded.length - 1];
981   _tOwned[account] = 0;
982   _isExcluded[account] = false;
983   _excluded.pop();
984
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 980

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
979   if (_excluded[i] == account) {
980   _excluded[i] = _excluded[_excluded.length - 1];
981   _tOwned[account] = 0;
982   _isExcluded[account] = false;
983   _excluded.pop();
984
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1171

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1170   address[] memory path = new address[](2);
1171   path[0] = address(this);
1172   path[1] = uniswapV2Router.WETH();
1173   _approve(address(this), address(uniswapV2Router), tokenAmount);
1174   uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
1175
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1172

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1171   path[0] = address(this);
1172   path[1] = uniswapV2Router.WETH();
1173   _approve(address(this), address(uniswapV2Router), tokenAmount);
1174   uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
1175   tokenAmount,
1176
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1380

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1379   if (
1380   _rOwned[_excluded[i]] > rSupply ||
1381   _tOwned[_excluded[i]] > tSupply
1382   ) return (_rTotal, _tTotal);
1383   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1384
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1381

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1380   _rOwned[_excluded[i]] > rSupply ||
1381   _tOwned[_excluded[i]] > tSupply
1382   ) return (_rTotal, _tTotal);
1383   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1384   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1385
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1383

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1382   ) return (_rTotal, _tTotal);
1383   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1384   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1385   }
1386   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1387
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1384

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1383   rSupply = rSupply.sub(_rOwned[_excluded[i]]);
1384   tSupply = tSupply.sub(_tOwned[_excluded[i]]);
1385   }
1386   if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
1387   return (rSupply, tSupply);
1388
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1483

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1482   address[] memory path = new address[](2);
1483   path[0] = uniswapV2Router.WETH();
1484   path[1] = address(this);
1485
1486   // make the swap
1487
```

SWC-110 | OUT OF BOUNDS ARRAY ACCESS

LINE 1484

low SEVERITY

The index access expression can cause an exception in case of use of invalid array index value.

Source File

- BUNNYVERSE.sol

Locations

```
1483   path[0] = uniswapV2Router.WETH();
1484   path[1] = address(this);
1485
1486   // make the swap
1487   uniswapV2Router.swapExactETHForTokensSupportingFeeOnTransferTokens{value:
bnbAmountInWei}(
1488
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 865

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BUNNYVERSE.sol

Locations

```
864   swapAndLiquifyEnabled = true;
865   tradingActiveBlock = block.number;
866   deadBlocks = _deadBlocks;
867   }
868
869
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1024

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BUNNYVERSE.sol

Locations

```
1023   ){
1024   if(tradingActiveBlock > 0 && (tradingActiveBlock + deadBlocks) > block.number){
1025   _isSniper[to]=true;
1026   }
1027
1028
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1036

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BUNNYVERSE.sol

Locations

```
1035   if (to != owner() && to != address(uniswapV2Router) && to !=
address(uniswapV2Pair)){
1036   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1037   _holderLastTransferTimestamp[tx.origin] = block.number;
1038   }
1039   }
1040
```

SWC-120 | POTENTIAL USE OF "BLOCK.NUMBER" AS SOURCE OF RANDOMNESS.

LINE 1037

low SEVERITY

The environment variable "block.number" looks like it might be used as a source of randomness. Note that the values of variables like coinbase, gaslimit, block number and timestamp are predictable and can be manipulated by a malicious miner. Also keep in mind that attackers know hashes of earlier blocks. Don't use any of those environment variables as sources of randomness and be aware that use of these variables introduces a certain level of trust into miners.

Source File

- BUNNYVERSE.sol

Locations

```
1036   require(_holderLastTransferTimestamp[tx.origin] < block.number, "_transfer::
Transfer Delay enabled.  Only one purchase per block allowed.");
1037   _holderLastTransferTimestamp[tx.origin] = block.number;
1038   }
1039   }
1040
1041
```

DISCLAIMER

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to, or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without Sysfixed's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Sysfixed to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model, or legal compliance.

This is a limited report on our findings based on our analysis, in accordance with good industry practice as of the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Sysfixed and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (Sysfixed) owe no duty of care.

ABOUT US

Sysfixed is a blockchain security certification organization established in 2021 with the objective to provide smart contract security services and verify their correctness in blockchain-based protocols. Sysfixed automatically scans for security vulnerabilities in Ethereum and other EVM-based blockchain smart contracts. Sysfixed a comprehensive range of analysis techniques—including static analysis, dynamic analysis, and symbolic execution—can accurately detect security vulnerabilities to provide an in-depth analysis report. With a vibrant ecosystem of world-class integration partners that amplify developer productivity, Sysfixed can be utilized in all phases of your project's lifecycle. Our team of security experts is dedicated to the research and improvement of our tools and techniques used to fortify your code.